# **Project Report for**

# **Machine Learning Group Project**

# 

## **Master’s in Data Science and Advanced Analytics at NOVA IMS, Lisbon**

## **Group Information**

Group: MAA\_202021\_50

Group members: **(Everyone please check and correct the names)**

Md. Shawkatul Islam Aziz

Md. Tahir Hossain

Philipp Metzger

Ali Sabbir

Henrique Vaz

**Abstract**

((Summarise the introduction))

This project’s goal is to implement Machine Learning methods and algorithms for classifying individuals as having an income below or above average. For this, with a training dataset that was provided, multiple techniques are implemented and tested.

((Summarise the methodology))

In order to complete the task an exploratory analysis of the training dataset is conducted, several transformations are applied to certain variables, the existence of missing values is assessed, the discriminatory power of categorical features is analysed, the categorical variables are encoded, an outlier detection is performed, the features are scaled and an appropriate subset of the original features is selected for further processing. Then, ((…))

((Summarise the Results))

((Summarise the Conclusions))

**I. INTRODUCTION**

The goal of this project is to implement Machine Learning algorithms that given the suitable input data are capable of predicting if an individual has an income lower or higher than the average income in a group of citizens.

For this, a dataset of 22400 observations serves as training data. This dataset includes amongst others the variables ‘Birthday’, ‘Marital Status’ and ‘Education Level’ and the binary target variable ‘Income’. A full list of the variables contained in the dataset will be presented in Chapter III. The input data and target vector are used to train several predictive models of which one is chosen in the end to be the best suited model for the task.

The project is related to the Kaggle competition ‘Newland’ in which several groups of students of the course Data Science and Advanced Analytics at the Lisbon based university NOVA IMS compete in a competition that is embedded in the fictitious scenario of the colonisation of a newly discovered habitable planet. In this Kaggle competition, each group uploads a vector of predictions computed with their best model, based on the input data of a test dataset provided in the project materials. The vector of predictions serves as the quality measure to assess which group designed and implemented the best predictive model.

**II. BACKGROUND**

In this chapter, the theoretical background of the techniques or algorithms who have not been explored during the practical classes but are applied in this project are explained.

**Support Vector Machine**

Support vector machines (SVM) are a set of supervised learning methods used for classification, regression and outliers detection. It is also a highly effective dimensional space. Still effective in cases where the number of dimensions is greater than the number of samples.Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient. Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels. We tried to find the best parameters using grid search, and created the GridSearch instance with selected parameters.

**Random Oversampling**

When one group of a categorical or binary target variable has more observations than the others (in the categorical case) or the other (in the binary case), it can be sensible to use random oversampling to harmonise the number of observations associated to each category of the variable. In random oversampling this is achieved by duplicating observations that are associated to the category or the categories that originally have less observations associated to them. The observations to duplicate are selected randomly and with repetition. The goal of random oversampling can be that after the process, all categories have the same number of observations associated with them. Or in the binary case, it is also be an option to define a factor, by which the numbers of observations of the two classes differ from each other such that n\_small = a \* n\_large, ∈ IR, a ∈ (0,1], where n\_small is the number of observations associated with the class that originally had less observations and n\_large is the number of observations associated with the other class. Random oversampling is described in Chapter 5 (Data Level Preprocessing Methods) of [1].

**III. METHODOLOGY** ((check the order of everything in the notebook))

The materials used in this project are the training dataset and the test dataset provided. The original training dataset consists of 22400 observations and 15 variables (including the target variable). These variables are shown in table 1 which is from the file ‘Project Presentation.pdf’ which serves for defining the fictitious background and the goal of the project.

|  |  |
| --- | --- |
| **Variable** | **Description** |
| Citizen\_ID | Unique identifier of the citizen |
| Name | Name of the citizen (First name and surname) |
| Birthday | The date of Birth |
| Native Continent | The continent where the citizen belong in the planet Earth |
| Marital Status | The marital status of the citizen |
| Lives with | The household environment of the citizen |
| Base Area | The neighborhood of the citizen in Newland |
| Education Level | The education level of the citizen |
| Years of Education | The number of years of education of the citizen |
| Employment Sector | The employment sector of the citizen |
| Role | The job role of the citizen |
| Working Hours per week | The number of working hours per week of the citizen |
| Money Received | The money payed to the elements of Group B |
| Ticket Price | The money received by the elements of Group C |
| Income | The dependent variable (Where 1 is Income higher than the average and 0 Income Lower or equal to the average) |

The software that is used in order to complete the project is Python and more precisely Anaconda and Jupyter Notebook. In the latter the code for this project is created.

In the following part, the steps conducted in our Jupyter notebook are described.

First, all packages and libraries that are used are imported. These include amongst others libraries from the standard Python library such as ‘os’ and packages from ‘sklearn’ which are used for the predictive models but also the library ‘xgboost’ which we used for feature selection, but not for classification.

In the second step the data described above is loaded into main memory and a first data exploration is conducted through which it becomes clear, that the training dataset contains more observations with the target variable ‘Income’ equal to 0 than observations with the target variable ‘Income’ equal to 1, meaning that in the training dataset, there are more observations that are classified as having an income lower than the average.[[1]](#footnote-1)

In the following step the variable ‘Birthday’ is transformed to obtain the age of each individual. As the unit for this newly created variable days is chosen and the old variable is replaced by the new one.

In the next step, the existence of cells containing empty strings, spaces or NaN values[[2]](#footnote-2) is assessed.

In a further exploratory analysis, the discriminatory power of the categorical features in the training dataset is assessed by plotting bar charts for each category in which the affiliation the either target value is represented in either grey or green.

The following step’s purpose is to one-hot encode most of these categorical features in order to make it possible to process them in the algorithms used later on. Only the feature ‘Education Level’ is not one hot encoded. For this one, a different encoding method is designed and applied, which assigns a numerical rank to each original value of this variable.

Next, outlier detection is performed. For this, the minima and maxima of features that could potentially have outliers are computed and and assessment is made whether or not these values are realistic. The result of this assessment is described in chapter IV.

The following step is the procedure of feature scaling. For this, a standard scaler is used, which standardises every feature by removing its mean and scaling it to unit variance [2].

Next, as a first feature selection step, correlations between the metric variables as well as between all variables are assessed to decide, which features are contributing only redundant information to the dataset. The outcome of this analysis is described in more detail in chapter IV; for now it is relevant to know that some features are discarded and a list of features ‘features\_to\_keep\_1’ is defined, which contains the features that are kept after the correlation analysis.

In a further feature selection step, RFE, a Ridge classifier and XGBoost are applied in order to identify the most important features. The result of this further limitation of the feature space is saved in the list ‘features\_to\_keep\_2’. As it is explained in Chapter IV, in the course of the project, the decision was made to use the features in ‘features\_to\_keep\_1’ for all models and to disregard the limitations of the feature space suggested by this step.

((missing: everything including and below ‘**Feature selection based on MLP classifier’))**

((Ali Sabbir from here:))

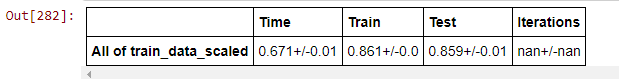
IV. RESULTS   
**1. Feature Selection based on MLP Classifier**

After detailed data processing and feature engineering, we have selected 5 features as feature\_certainly\_to\_keep, which accuracy for train 0.831, test 0.829. If we consider 1 feature to keep, the model accuracy for train 0.867, test 0.852 and even in all features selection our model is overfitting and no increase in test accuracy.

### 

## **2. Feature selection based on AdaBoost classifier**

According to Feature selection based on AdaBoost classifier keeping the same features from previous selection, model has accuracy for train 0.837, test 0.838 in features\_certainly\_to\_keep. And for 1 feature to keep, the model accuracy for train 0.858, test 0.857 and even in all features selection our model has no significant improvement of the model after trying all features with train\_data\_scaled**.**

****

**3. Grid search AdaBoost (train\_data\_scaled)**

Initializing the model with random parameters using SAMME.R & SAMME algorithm, we have got the best parameter set {'algorithm': 'SAMME.R', 'learning\_rate': 0.95, 'n\_estimators': 500, 'random\_state': 42}. Step by step going deep into the process to look for the best parameters, we finally have {'algorithm': 'SAMME.R', 'learning\_rate': 0.96, 'n\_estimators': 508, 'random\_state': 42} with accuracy f1 0.867.



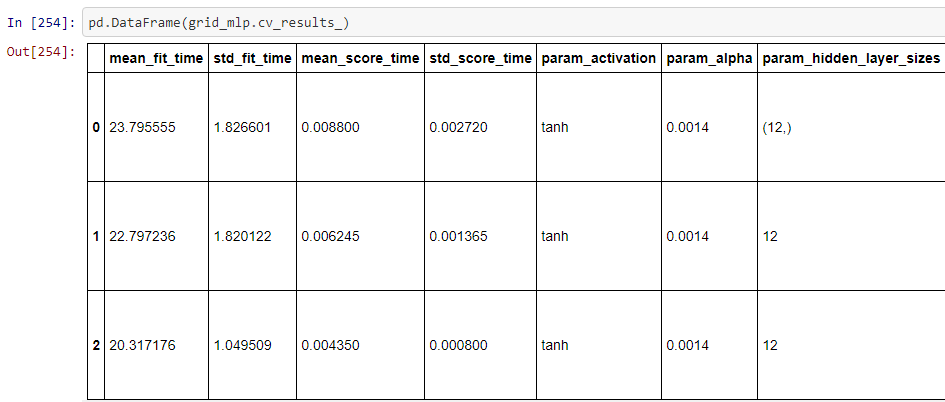
## **Train\_data\_scaled [features\_to\_keep\_1]**

The model Grid search AdaBoost as features to keep 1 with random parameters using algorithm SAMME.R & SAMME and got the best parameter set {'algorithm': 'SAMME.R', 'learning\_rate': 0.6, 'n\_estimators': 800, 'random\_state': 42} and after that we got even more closely final result {'algorithm': 'SAMME.R', 'learning\_rate': 0.61, 'n\_estimators': 962, 'random\_state': 42} with the best parameter f1 0.867 that close to train data scaled.

****

# **Create a GridSearch instance**

we got GridSearch instance with parameters set {'activation': 'tanh','alpha': 0.001, 'hidden\_layer\_sizes': (12,), 'learning\_rate': 'constant', 'learning\_rate\_init': 0.001, 'random\_state': 42, 'solver': 'adam'} and best possible out come from this parameter when **even finer (Check Alpha)** {'activation': 'tanh', 'alpha': 0.0014, 'hidden\_layer\_sizes': (12,), 'learning\_rate': 'constant', 'learning\_rate\_init': 0.001, 'random\_state': 42, 'solver': 'adam'} and best parameter f1 is 0.853125.

****

****

## 

## 

## Model selection

We will try different models, and for each model we will also apply different parameters. In the end we will check which model gives us better insights within each model's best parameter choice collection.

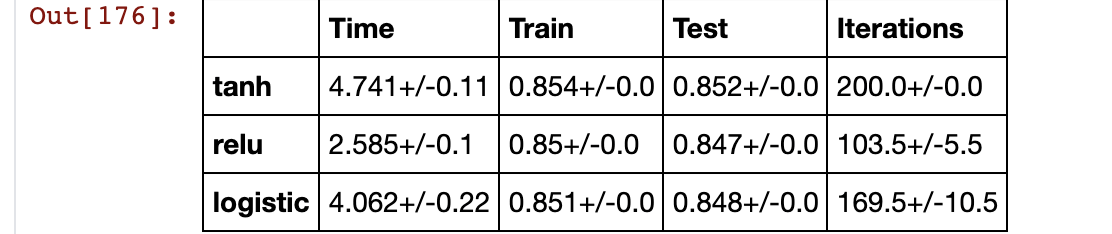
**MULTILAYER PERCEPTRON (MLP)**

we have been calculate it with random oversampling and without random oversampling where hidden layer size based on the number of hidden neurons should be between the size of the input layer and the size of the output layer and number of hidden neurons should be 2/3 the size of the input layer, plus the size of the output layer and the number of hidden neurons should be less than twice the size of the input layer.

We have done our model with several activation functions for the best hidden layer choice we got before.The functions we will use are:

* tanh
* logistic
* reLU (is the default one)

We sould for now keep the tanh activation function since it's the one that gives us a good balance between accuracy and time.



There are two type of learning rate adaptive and constant and used three parameters 0.001, 0.01, 0.1 with three solvers

* L-BFGS
* ADAM
* SGD

V. DISCUSION

VI. CONCLUSION

VII. REFERENCES

[1] Fernández, Alberto & García, Salvador & Galar, Mikel & Prati, Ronaldo & Krawczyk, Bartosz & Herrera, Francisco. (2018). Learning from Imbalanced Data Sets. 10.1007/978-3-319-98074-4.

[2] <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html>, viewed on 25. Dec. 2020 at 13:07

1. Income = 1: 76.29 %, Income = 1: 23.71 % [↑](#footnote-ref-1)
2. NaN stands for Not a Number, meaning that the respective value is non-existent. [↑](#footnote-ref-2)